SAKE*: A Symmetric Authenticated Key Exchange Protocol With Perfect Forward Secrecy for Industrial Internet of Things

Article *in* IEEE Transactions on Industrial Informatics · January 2022

DOI: 10.1109/TII.2022.3145584

5 authors, including:

Jianhua Chen
Chinese Academy of Agricultural Sciences

Mohammad Shojafar
University of Surrey

Saru Kumari
Chaudhary Charan Singh University

Debiao He
Wuhan University

SAKE*: A Symmetric Authenticated Key Exchange Protocol with Perfect Forward Secrecy for Industrial Internet of Things

Qing Fan, Jianhua Chen, Mohammad Shojafar, *Senior Member, IEEE,* Saru Kumari, and Debiao He, *Member, IEEE*

***Abstract*—Security in the Industrial Internet of Things (IIoT) is vital as there are some cases where IIoT devices collect sensory information for crucial social production and life. Thus, designing secure and efficient communication channels is always a research hotspot. However, end devices have memory, computation, and power-supplying capacities limitations. Moreover, Perfect Forward Secrecy (PFS) which means long-term key exposure still disclose previous session keys is a critical security property for Authentication and Key Exchange (AKE). This paper proposes an AKE protocol named SAKE* for the IIoT environment, where two types of keys (i.e., a master key and an evolution key) guarantee PFS. In addition, the SAKE* protocol merely uses concatenation, XOR, and hash function operations to achieve lightweight authentication, key exchange, and message integrity. We also compare the SAKE* protocol with seven current and IoT-related authentication protocols regarding security properties and performance. Comparison results indicate that the SAKE* protocol consumes the least computation resource and third least communication cost among eight AKE protocols while equipping with twelve security properties.**

***Index Terms*—Symmetric key, perfect forward secrecy, authentication and key exchange, Industry 4.0.**

I. Introduction

The combination of the Internet of Things (IoT) with an industrial ecosystem, also referred to as Industrial Internet of Things (IIoT), has vitalized the concept of the fourth industrial revolution (Industry 4.0) [1]. There are plenty of areas where critical functions, such as automated manufacturing [2], intelligent agriculture [3] and smart transportation [4], specifically rely on IIoT sensory data. According to a report by the IBM institute, IIoT could add 14 trillion USD to the global economy by 2030 [5]. Nevertheless, in some cases, any slight deviation of accurate data may cause tremendous harm to personal life or social order [6], including but not limited to IoT-based patient-health monitoring and vehicle traffic control. As a consequence, data transmission must be achieved in a secure, and authenticated manner [7].

Q. Fan, J. Chen and School of Mathematics and Statistics Wuhan University Wuhan, China (e-mail: qingfan_cassie.whu@whu.edu.cn, chenjh_ecc@163.com)

M. Shojafar is with the 5GIC & 6GIC, Institute for Communication Systems (ICS), University of Surrey, Guildford, GU27XH, United Kingdom (e-mail: m.shojafar@surrey.ac.uk)

Saru Kumari is with the Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India (e-mail:saryusiirohi@gmail.com).

D. He is with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China, and also with the Shanghai Key Laboratory of Privacy-Preserving Computation, MatrixElements Technologies, Shanghai 201204, China (email: comhedebiao@163.com) (Corresponding author: Debiao He.)

Manuscript received 11 September 2021; revised 27 November 2021; accepted 18 January 2022.

IoT devices are resource-constraint in processing, memory, and batteries power availability from. Thus, these IoT devices use wireless as the intermediary of communication [8] which implies potential security and privacy issues. Avoine et al. [9] and Fang et al. [10] have devoted themselves to general IoT system architecture research and proposed the IoT trust model. Concretely, an IoT system involves three entities: the trusted Authority (TA), IoT devices, and users. The TA is responsible for the system's security, whose primary function is to help the IoT device and user realize mutual authentication and session keys agreement. The TA builds a trustful bridge between both parties and provides excellent effectiveness in establishing secure channels, which means the operating load of IoT devices can be alleviated in this trust model.

However, master secret keying material used for session key generation could be leaked in uncertain situations such as insider stealing and key medium broken. This may make previous session channels not secure. As a strong form of long-term security, perfect forward secrecy (PFS) [11] guarantees the desirable fundamental property of AKE protocol. It embodies that although the master key is revealed at some point, all the past communication channels are still secure. PFS facilitates the utilization of limited-life session keys in our proposed AKE protocol for IIoT environments. Many AKE protocols with PFS take advantage of digital signature [12], [13] or symmetric encryption for authentication, adopt the Diffie-Hellman (DH) exchange for session key agreement and select different random values in each session to ensure PFS [14], [15]. However, such protocols put a great demand on computing and storage capacities and reduce the usage life of power-limited end devices.

Bellare and Rogaway [16] and Avoine et al. [17] have proposed symmetric-key AKE protocols while these protocols either neglect PFS or lack the TA participation. We take into account the various constraints and requirements of IIoT scenarios. Then we use two lightweight primitives: pseudo-random function (PRF) and messages authentication code (MAC) [18], [19] for authentication and key negotiation. The proposed protocol in this paper seeks to minimize the calculated quantity of both parties through allocating authentication and session freshness work to the TA.

There are two types of keys in our proposed SAKE* protocol: a secret symmetric master key denoted as K shared between authenticated two sides, and an evolution key denoted as K_α shared among TA and authenticated parties (i.e., user A and IoT node B) where $\alpha \in \{A, B\}$ denotes entities which hold the evolution key. The TA uses a pair of evolution keys to judge the synchronization state and realize the legality authentication of both sides. At the same time, both parties use K to compute actual session keys based on the current evolution key authentication. After one session completion, K_α and K have been updated at least once. An attacker cannot acquire the previous session key even if it gains the current master key K owing to the one-wayness of the update function. The proposed SAKE* protocol also overcomes diverse attacks under Brzuska et al.'s model [20] and the IIoT system's security is difficult to break.

In summary, we propose an AKE system model and design a new AKE protocol with perfect forward secrecy for IIoT environments named SAKE*. The proposed protocol is sound (whose definition is given in section IV-A) and provably secure under Brzuska et al.'s model. Comparisons with seven current and IoT-related AKE protocols demonstrate that our protocol is lightweight, and has better security properties and performance.

The rest of this paper is organized as follows. Preliminaries including system model and security model are in section II. Section III proposes SAKE* protocol consisting of initialization phase and AKE phase. Soundness, security proof and security analysis are in section IV. Section V analyze SAKE*'s security and performance by comparing it with seven related protocols. Finally, section VI gives this paper's conclusion.

II. Preliminaries

A. The System Model

In this section, we propose the SAKE* system model as illustrated in figure 1, which is composed of three entities: Trust Authority (TA), the IoT end device (ED), and the User.

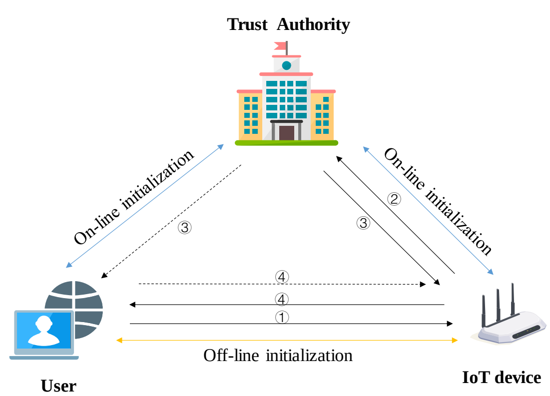


Fig. 1. System Model of SAKE*

- **Trust Authority**: TA has great computing capacity to help establish secure channels. In the proposed SAKE*, TA is responsible for generating, distributing, and managing evolution keys. In addition, TA utilizes held evolution keys to authenticate IoT end devices and users.
- **Users**: For requesting data from EDs, the resource-limited user sends authentication messages to the corresponding IoT device. Then he computes the session key and completes the master key update, having verified authentication messages from ED or TA.
- **IoT end devices**: It is responsible for sensing and collecting information from specific environments. In the SAKE* protocol, EDs accomplish authentication, session key generation, and master key update.

In our proposed SAKE* system model, the user/IoT end device first performs on-line initialization with TA and off-line initialization. When the user tries to build a secure channel with some ED, he sends his identity, a random value, and the hiding evolution key information to ED (Step ①). Then the ED transmits his identity and hiding evolution key together with those of the user's to TA (Step ②). Upon receiving messages from the IoT end device, TA verifies reliability through the masked evolution key and determines the synchronized state of two parties. Suppose the ED is one step behind the user. In that case, TA sends related messages to ED (Step ③ on solid line), and ED transmits the final authenticated messages to the user (Step ④ on solid line). Otherwise, related messages are sent to the user (Step ③ on the dotted line), and the eventual messages are sent to ED (Step ④ on the dotted line). The TA's selection process of target communicator in step ③ guarantees the soundness of SAKE*.

B. Design Objectives

We try to design a symmetric-key authenticated key exchange protocol with perfect forward secrecy for IoT environments to provide security and privacy protection against the threat model. Referring to researches in [10], [21], the design targets of our proposed SAKE protocol are as follows.

- **Mutual authentication**: Any two parties who attempt to set secure channels should authenticate each other to ensure that the entity is the one claimed. The attacker could not pretend to be a legitimate entity to send malicious messages by impersonation attacks.
- **End-to-end security**: The authenticated secret information transmitted in the AKE protocol could only be read by the targeted entities.
- **Authenticated data integrity**: The data that is sent to both authenticated parties and utilized to authentication and key agreement cannot be altered during transmission in the proposed AKE protocol.
- **Secure session key agreement**: The session key should only be exchanged between two authenticated parties to secure communication in the IoT environment.
- **Perfect forward secrecy**: Disclosure of the master key will not compromise the previous session key. That is, previous secure data transmission channels are still confidential.
- **Soundness**: This property requires that when a valid session finishes, authenticated entities in the AKE phase reach the synchronized state, that is master key used to compute the session key has been updated at least once, and the negotiated session key is correct and consistent.
- **The known attacks resistance**: For a secure communicational environment, the proposed AKE protocol should resist known attacks, such as impersonation attacks, man-in-the-middle attacks, replay attacks, eavesdropping attack, and data tampering attacks.
- **Lightweight resource overload**: On the premise of satisfying security requirements, the protocol's rounds,

computation cost, and communication cost should be as least as possible.

III. Proposed SAKE* Protocol

In this section, we detailedly illustrate the proposed SAKE* protocol from aspects of the initialization phase and authentication and key exchange phase.

A. Initialization

Initialization of the proposed SAKE* protocol consists of on-line initialization and off-line initialization. The On-line initialization phase is realized by the user (U_i) and server (TA), when U_i gets the evolution key K_i, component element R_i of exchange key as well as a random number r_i. Specifically, evolution key K_i is calculated by TA with his secret key s and message from U_i, which is undoubtedly in sync with master key K. Off-line initialization phase is executed by two authenticated entities U_A and IoT_B when two users obtain the initialized master key K. Two hash functions $h : \{0,1\}^* \rightarrow \{0,1\}^k$ and $H : \{0,1\}^* \rightarrow F_p^n$, where k is the security parameter, F_p^n denotes n dimensional column vector with elements in finite field F_p, are used in this scheme. Details of these two phases are described as follows.

- **On-line Initialization**:
  - Authenticated entities $\alpha(\in \{A, B\})$ first inputs a password PW_α and generates a random number θ_α by the random number generator. Then he uses the identity ID_α, password PW_α and random θ_α to compute $g_\alpha = h(ID_\alpha||PW_\alpha||\theta_\alpha)$ and $h_\alpha = h(ID_\alpha||PW_\alpha||g_\alpha)$. $\{ID_\alpha, g_\alpha, h_\alpha\}$ are sent to the TA through a secure channel.
  - Then TA uses random number generator to produce the secret key s and an $n \times m(n > m)$ non-singular matrix M with elements in F_p. Then he computes $K_\alpha^0 = h(s||g_\alpha)$, $R_\alpha = H(ID_\alpha||h_\alpha)^T \cdot M$ and stores $(ID_\alpha, \perp, g_\alpha, K_\alpha^0)$. Finally, the TA sends $\{K_\alpha^0, R_\alpha\}$ to α.
  - α sets the received K_α^0 as K_α and secretly stores $\{K_\alpha, R_\alpha, \theta_\alpha\}$ in a confidential and hard-to-be-stolen device.
- **Off-line Initialization**:
  As long as two different entities U_A (resp. IoT_B) receives the other one's identity ID_B (resp. ID_A), element R_B (resp. R_A) and a trusted certificate C_B (resp. C_A), U_A (resp. IoT_B) computes $g_A = h(ID_A||PW_A||\theta_A)$, $h_A = h(ID_A||PW_A||g_A)$ and the initialized master key $K = R_B \cdot H(ID_A||h_A)$ (resp. $g_B = h(ID_B||PW_B||\theta_B)$, $h_B = h(ID_B||PW_B||g_B)$ and $K = R_A \cdot H(ID_B||h_B)$) off-line.

B. Authentication and Key Exchange

The meaning of some notations used in this phase can be seen as follows.

- kdf corresponds to: $sk = KDF(K, ID_A||ID_B||f(r_A, r_B))$
- upd_A corresponds to:
  1) $K_A^{j-2} \leftarrow K_A^{j-1}$
  2) $K_A^{j-1} \leftarrow K_A^j$
  3) $K_A^j \leftarrow h(s||g_A||K_A^{j-1})$
- upd_B corresponds to:
  1) $K_B^{j-2} \leftarrow K_B^{j-1}$
  2) $K_B^{j-1} \leftarrow K_B^j$
  3) $K_B^j \leftarrow h(s||g_B||K_B^{j-1})$

When an IoT user U_A/IoT_B initiates a new confidential communication, they first execute authentication and key exchange to negotiate a new session key with perfect forward secrecy under the help of trust authority TA.

Before launching this protocol, each user holds **his current** master key K and corresponding evolution key $K_\alpha(\alpha \in \{A, B\})$. Similarly, TA also keeps identities with evolution keys of three periods ($ID_\alpha, K_\alpha^{j-2}, K_\alpha^{j-1}, K_\alpha^j$), where j denotes the **current** phase of TA. In the end of this phase, U_A and IoT_B complete at least one update of K. We assume that U_A is the initiator of SAKE* protocol.

- U_A first uses his password PW_A, identity ID_A and θ_A to compute the secret g_A. Then he uses evolution key K_A to hide g_A. Meanwhile, phase information of evolution is contained in τ_A, that is $\tau_A = g_A \oplus K_A$. Finally, U_A sends a random $r_A \in \{0,1\}^k$ used to mark this session and ID_A, τ_A to the target communicator.

Algorithm 1 U_A side

Input:
$ID_A, PW_A, \theta_A, K_A$;
Output:
τ_A, r_A;
1: Compute $g_A = h(ID_A||PW_A||\theta_A)$;
2: $\tau_A = g_A \oplus K_A$;
3: $r_A \in_R \{0,1\}^k$;
4: Send $\{ID_A, r_A, \tau_A\}$ to IoT_B.

- Upon receiving messages from U_A, IoT_B firstly verifies this session's freshness. It should be clear that IoT_B holds one random number of U_A's last session request to avoid replay attack. Then IoT_B performs similar computation as U_A, that is computes $g_B = h(ID_B||PW_B||\theta_B)$, $\tau_B = g_B \oplus K_B$ and generates a random number $r_B \in \{0,1\}^k$. Finally,he sends $\{ID_B, \tau_B, r_B\}$ together with $\{ID_A, \tau_A\}$ to TA.
- Since TA holds a list of users' authentication information, he could judge the synchronization state by evolution keys of U_A and IoT_B. Upon receiving two identities with corresponding τ_A, τ_B, TA first estimates the gap of evolution keys between that held by U_A (resp. IoT_B) and that held by TA.
  - If evolution keys of U_A and IoT_B keep pace with that held by TA, TA assigns the current K_A^j, K_B^j respectively to K_1, K_2 which will be used to compute messages authentication code (MAC). Then TA updates once evolution keys i.e., upd_A, upd_B for U_A, IoT_B. In this condition, δ_{AB} is set 0 and $\epsilon_A = \epsilon_B = 0$.

Algorithm 2 IoT_B side

Input:
$ID_A, ID_B, PW_B, \theta_B, K_B$;
Output:
τ_B, r_B;
if r_A is not fresh **then**
abort;
else
Compute $g_B = h(ID_B||PW_B||\theta_B)$;
$\tau_B = g_B \oplus K_B$;
$r_B \in_R \{0,1\}^k$;
end if
Send $\{ID_B, \tau_B, r_B, ID_A, \tau_A\}$ to TA.

- If evolution keys of both U_A and IoT_B are one step behind that of TA, TA assigns previous stage evolution keys K_A^{j-1}, K_B^{j-1} respectively to K_1, K_2. In this case, TA is unnecessary to updates evolution keys upd_A, upd_B since his current evolution keys K_A^j, K_B^j are updated keys in view of U_A and IoT_B. In this condition, δ_{AB} is set 0 and $\epsilon_A = \epsilon_B = 0$.
- If evolution key of U_A is one step behind those of IoT_B and TA, TA sets $K_1 = K_A^{j-1}, K_2 = K_B^j$ and updates once K_A, K_B. In this condition, $\delta_{AB} = -1, \epsilon_A = 1, \epsilon_B = 0$.
- If evolution key of IoT_B is one step behind those of U_A and TA, TA sets $K_1 = K_A^j, K_2 = K_B^{j-1}$, updates once K_A, K_B. In this condition, $\delta_{AB} = 1, \epsilon_A = 0, \epsilon_B = 1$.

Eventually, TA uses g_A (resp. g_B) to encrypt K_A^j (resp. K_B^j) and uses K_1 (resp. K_2) to compute MACs. If $\epsilon_A = 0$ (i.e., $\delta_{AB} \neq -1$), TA sends $\{ID_B, \epsilon_B, S_B, \tau_{CB}\}$ and $\{ID_A, \epsilon_A, S_A, \tau_{CA}\}$ to IoT_B; otherwise, TA sends $\{ID_B, \epsilon_B, r_B, S_B, \tau_{CB}\}$ and $\{ID_A, \epsilon_A, S_A, \tau_{CA}\}$ messages to U_A.

Case 1 ($\delta_{AB} \neq -1$):

- Upon receiving messages from TA, IoT_B utilizes his current K_B to verify whether $Mac(K_B||\epsilon_B||S_B \oplus g_B) = \tau_{CB}$. It aborts if the verification fails; otherwise, IoT_B assigns $S_B \oplus g_B$ to K_B.
  * If $\epsilon_B = 1$, IoT_B updates K once and computes session key sk. Then IoT_B updates K once again.
  * If $\epsilon_B = 0$, IoT_B computes session key sk and updates K once.

  Finally, IoT_B calculates τ_{BA}= $Mac(K$,$r_A||r_B||sk)$ and sends $\{\epsilon_A$,S_A, τ_{CA},$ID_B, r_B, \tau_{BA}\}$ to U_A.
- Upon receiving messages from IoT_B, U_A uses his current K_A to verify whether $Mac(K_A||\epsilon_A||S_A \oplus g_A) = \tau_{CA}$. It aborts if the verification fails; otherwise, U_A assigns $S_A \oplus g_A$ to K_A. If $\epsilon_A = 0$, U_A computes session key sk and updates K once. Finally, U_A verifies whether sk is the same with that of IoT_B by $Vrf(K||r_A||r_B||sk, \tau_{BA})$.

Case 2 ($\delta_{AB} = -1$):

- Upon receiving messages from TA, U_A uses his current K_A to verify whether $Mac(K_A||\epsilon_A||S_A \oplus$

Algorithm 3 TA side

Input:
$ID_A, ID_B, K_A^{j-2}, K_A^{j-1}, K_A^j, K_B^{j-2}, K_B^{j-1}, K_B^j, s, \tau_A, \tau_B, r_B$;
Output:
$\epsilon_A, S_A, \tau_{CA}, \epsilon_B, S_B, \tau_{CB}$;
if $h(s||\tau_A \oplus K_A^j||K_A^{j-1}) = K_A^j \& h(s||\tau_B \oplus K_B^j||K_B^{j-1}) = K_B^j$ **then**
$K_1 = K_A^j, K_2 = K_B^j$, upd_A, upd_B;
$\delta_{AB} = 0, \epsilon_A = 0, \epsilon_B = 0$;
else $\{h(s||\tau_A \oplus K_A^{j-1}||K_A^{j-2}) = K_A^{j-1} \& h(s||\tau_B \oplus K_B^{j-1}||K_B^{j-2}) = K_B^{j-1}\}$
$K_1 = K_A^{j-1}, K_2 = K_B^{j-1}$;
$\delta_{AB} = 0, \epsilon_A = 0, \epsilon_B = 0$;
else $\{h(s||\tau_A \oplus K_A^{j-1}||K_A^{j-2}) = K_A^{j-1} \& h(s||\tau_B \oplus K_B^j||K_B^{j-1}) = K_B^j\}$
$K_1 = K_A^{j-1}, K_2 = K_B^j$, upd_A, upd_B;
$\delta_{AB} = -1, \epsilon_A = 1, \epsilon_B = 0$;
else $\{h(s||\tau_A \oplus K_A^j||K_A^{j-1}) = K_A^j \& h(s||\tau_B \oplus K_B^{j-1}||K_B^{j-2}) = K_B^{j-1}\}$
$K_1 = K_A^j, K_2 = K_B^{j-1}$, upd_A, upd_B;
$\delta_{AB} = 1, \epsilon_A = 0, \epsilon_B = 1$;
else
abort.
end if
$S_A = K_A^j \oplus g_A, S_B = K_B^j \oplus g_B$;
$\tau_{CA} = Mac(K_1||\epsilon_A||K_A^j||S_A||ID_A)$;
$\tau_{CB} = Mac(K_2||\epsilon_B||K_B^j||S_B||ID_B)$;
if ϵ_A=0 **then**
Send $\{ID_A, \epsilon_A, S_A, \tau_{CA}, ID_B, \epsilon_B, S_B, \tau_{CB}\}$ to IoT_B;
else
Send $\{ID_A, \epsilon_A, S_A, \tau_{CA}, ID_B, \epsilon_B, r_B, S_B, \tau_{CB}\}$ to U_A.
end if

Algorithm 4 IoT_B side (Case 1)

Input:
$K_B, \epsilon_B, S_B, g_B, \tau_{CB}, \tau_{CA}, K, ID_A, ID_B, r_A, r_B$;
Output:
τ_{BA}, sk, updated (K, K_B);
Compute $\tau'_{CB} = Mac(K_B||\epsilon_B||S_B \oplus g_B||S_B||ID_B)$;
if $\tau_{CB} \neq \tau'_{CB}$ **then**
abort;
else
$K_B = S_B \oplus g_B$;
end if
if $\epsilon_B = 1$ **then**
$update(K), kdf, update(K)$;
else $\{\epsilon_B = 0\}$
$kdf, update(K)$;
else
abort
end if
$\tau_{BA} = Mac(K, r_A||ID_B||r_B||sk)$;
Send $\{ID_B, r_B, \tau_{BA}, \epsilon_A, S_A, \tau_{CA}\}$ to U_A.

Algorithm 5 U_A side (Case 1)

Input:
$\tau_{CA}, \epsilon_A, S_A, g_A, \tau_{BA}, K, ID_A, ID_B, r_A, r_B$;
Output:
sk, updated (K, K_A) or $abort$;

```
Compute τ'_CA = Mac(K_A||ε_A||S_A ⊕ g_A||S_A||ID_A);
if τ'_CA ≠ τ_CA then
    abort;
else
    K_A = S_A ⊕ g_A;
end if
if ε_A = 0 then
    kdf, update(K);
else
    abort;
end if
if Vrf(K||r_A||ID_B||r_B||sk, τ_BA) = 0 then
    abort;
else
    accept.
end if
```

$g_A) = \tau_{CA}$. It aborts if the verification fails; otherwise, U_A assigns $S_A \oplus g_A$ to K_A. If $\epsilon_A = 1$, U_A updates K once, computes session key sk and updates K once again. Finally, U_A uses K to calculate $\tau_{AB}= Mac(K,r_A||r_B||sk)$ and sends $\{\epsilon_B,S_B,\tau_{CB},ID_A,r_A,\tau_{AB}\}$ to IoT_B.

- Upon receiving messages from U_A, IoT_B uses his current K_B to verify whether $Mac(K_B||\epsilon_B||S_B \oplus g_B) = \tau_{CB}$. It aborts if the verification fails; otherwise, IoT_B assigns $S_B \oplus g_B$ to K_B. If $\epsilon_B = 0$, IoT_B computes session key sk and updates K once. Finally, IoT_B verifies whether sk is the same with that of U_A by $Vrf(K||r_A||r_B||sk, \tau_{AB})$.

IV. Soundness and Security of SAKE*

We theoretically infer that (i) SAKE* is sound, and (ii) SAKE* is secure under Brzuska et al.'s model in this section.

A. Soundness

The soundness of SAKE* indicates that having finished a session key negotiation, both U_A and IoT_B respectively have updated their internal state, been in sync with TA, and shared the same session key sk [20].

***Theorem** 1:* Assume that U_A is the initiator and IoT_B is the target communicator of SAKE*. Let δ_{AB} be the gap between U_A and IoT_B related to the evolution keys (master key) of both parties. There are the following conditions:

- $\delta_{AB} \in \{-1, 0, 1\}$;
- No matter what the synchronization state of U_A and IoT_B at the beginning of a new SAKE* protocol, when the session is complete, it has
  - The master key K of IoT_B and U_A has been updated at least once;
  - U_A, IoT_B and TA keep synchronized concerning master key K;
  - U_A and IoT_B negotiate the same session key.

Proof 1: The messages sent during the session are numbered from 1 to 4. An $(c_A, c_B, c_S)-session$ is the session in which $c_A/c_B/c_S$ is the last message sent to $U_A/IoT_B/TA$. Let (c_A, c_B, c_S) be monotonically increasing counters initialized to 0 that follows evolution keys held by U_A, IoT_B and TA. The $\delta_{AB} = c_A - c_B$ is the gap between U_A and IoT_B related to the evolution of their secret keys.

The first time U_A and IoT_B execute SAKE* protocol, they are synchronized, that is $\delta_{AB} = 0$ and $(c_A, c_B, c_S) = (0, 0, 0)$. TA can verify U_A and IoT_B's synchronization state through τ_A and τ_B respectively with K_A and K_B. Thereby, TA computes $\delta_{AB} = 0$ and $\epsilon_A = \epsilon_B = 0$. Consequently, for each possible value (c_A, c_B, c_S), it has:

- After a $(0, 1, 0) - session$, $\delta_{AB} = 0$ and $(c_A, c_B, c_S) = (i, i, i)$;
- After a $(0, 1, 2) - session$, $\delta_{AB} = 0$ and $(c_A, c_B, c_S) = (i, i, i + 1)$;
- After a $(0, 3, 2)-session$, $\delta_{AB} = -1$ and $(c_A, c_B, c_S) = (i, i + 1, i + 1)$;
- After a $(3, 4, 2) - session$, $\delta_{AB} = 0$ and $(c_A, c_B, c_S) = (i + 1, i + 1, i + 1)$.

Having finished the SAKE* protocol, possible values of δ_{AB} is 0 or -1 i.e., $(c_A, c_B, c_S) = (i, i, i + 1)$ or $(c_A, c_B, c_S) = (i, i + 1, i + 1)$. Meanwhile, U_A and IoT_B have updated their evolution keys and master keys at least once according to c_A/c_B change. They are synchronized and compute the same session key using the same K, r_A, r_B. If objective factors may cause interruption at some point, it would start with $\delta_{AB} = 0, (c_A, c_B, c_S) = (i, i, i + 1)$ or $\delta_{AB} = -1, (c_A, c_B, c_S) = (i, i + 1, i + 1)$ in the next time. Then we could use the above method to analyze δ_{AB}, master key update, synchronization state and session key agreement, which will finally infer Theorem 1.

B. Security Model

In this section, we use the execution environment and security definitions for authenticated key exchange protocol in [22], [23].

Parties. An authenticated key exchange protocol is executed by a set of parties $\mathcal{P} = \{P_0, ..., P_{n-1}\}$ with trust authority TA. Each party has an associated long-term key lk, and each pair of parties exchanges a distinct key lk.

Instances. Each party can participate in multiple sequential executions of authenticated key exchange protocol. Since the proposed SAKE* protocol is proceeded based on evolution keys, parallel executions are not allowed, which otherwise may cause abortion of some executions. Each execution of protocol is said a session, in which an instance π_i^s embodies the specific process of protocol π. Some notations used to describe the security model are as follows.

- β: it represents the role in the session of a protocol execution and is either the initiator or the responder, i.e., $\beta \in \{init, resp\}$.

- pid: it is the identity of communication party in the instance π_i^s, i.e., $pid \in \mathcal{P}$.
- γ: it represents the state of the instance, i.e., $\gamma \in \{accepted, rejected, running, \perp\}$.
- τ: it is the state of session key $\pi_i^s.sk$, i.e., $\tau \in \{revealed, \perp\}$.
- sid: it is the identifier of a session.
- b: it is a binary bit randomly sampled from $\{0, 1\}$.

Adversarial Queries. According to Brzuska et al.'s model, we assume that the adversary $\mathcal{A}$ controls over the network and interacts with oracles by performing queries. Admissible queries in the AKE protocol are as follows.

- *NewSession(P_i, β, pid):* this query establishes a new instance π_i^s at P_i with the role β, in which the intended communicator identity is pid.
- *Send(π_i^s, m):* this query admits $\mathcal{A}$ to transmit message m to π_i^s. If $\pi_i^s.\gamma \neq running$, it returns $\perp$. Otherwise, π_i^s responds specific message as the protocol.
- *Corrupt(P_i):* If the u-th query performed by $\mathcal{A}$ is $Corrupt(P_i)$, P_i is called $u - corrupted$. For an entity that does not be corrupted, we set $u = +\infty$. Finally, this query returns the long-term key $P_i.lk$ of P_i.
- *Reveal(π_i^s):* if π_i^s has been accepted, this query returns the session key $\pi_i^s.sk$ and $\pi_i^s.\tau$ is set to $revealed$.
- *Test(π_i^s):* this query could be asked only once in this game. If $\pi_i^s.\beta \neq accepted$, it returns $\perp$. Otherwise, it selects a random bit $b \in \{0, 1\}$ and returns sk_b to $\mathcal{A}$ where sk_0 is sampled from key space $\mathcal{K}$ and $sk_1 = \pi_i^s.sk$.

***Definition** 1 (Matching Conversations):* We say that π_i^s has a matching conversation to π_j^t if

- All messages sent by π_i^s are received by π_j^t;
- All messages sent by π_j^t are received by π_i^s.

***Definition** 2 (Freshness):* A session π_i^s is called to be fresh with intended partner P_j, if

- When $\mathcal{A}$ runs its u_0-th query, $\pi_i^s.\gamma = accepted$ and $\pi_i^s.pid = P_j$;
- $\pi_i^s.\tau \neq revealed$ and P_i is $u - corrupted$ with $u > u_0$;
- for any partner instance π_j^t of π_i^s, it has that $\pi_j^t.\tau \neq revealed$ and P_j is $u' - corrupted$ with $u' > u_0$.

It can be noted that *freshness* captures the property of forward secrecy.

***Definition** 3 (Secure Entity Authentication):* A protocol π_i^s is called to be maliciously accepted with targeted partner P_j in the security experiment when it has

- $\pi_i^s.\gamma = accepted$ and $\pi_i^s.pid = P_j$ in $\mathcal{A}$'s u_0-th query;
- P_i and P_j are uncorrupted (resp. u- and u'-corrupted with $u, u' > u_0$);
- there is no unique session π_j^t such that π_i^s and π_j^t are partners.

We assign $Adv_{\Pi}^{auth}(\mathcal{A})$ denote the probability that the adversary $\mathcal{A}$ makes a session accept maliciously in the AKE security experiment.

***Definition** 4 (Key Indistinguishability):* An adversary $\mathcal{A}$ that issued *Test* query to session π_i^s in the AKE security experiment, answers *Test* query correctly if it aborts with outputting correct b' that satisfies

- π_i^s is fresh with targeted partner P_j;
- $\pi_i^s.b = b'$.

We let $Adv_{\Pi}^{key-ind}(\mathcal{A})$ denote advantage to the event that $\mathcal{A}$ answers the *Test* query correctly and it has

$$Adv_{\Pi}^{key-ind}(\mathcal{A}) = |Pr[\pi_i^s.b = b'] - \frac{1}{2}|$$

***Definition** 5 (AKE Security):* An AKE protocol Π is said to be secure if for all probabilistic polynomial time (PPT) adversary $\mathcal{A}$, $Adv_{\Pi}^{auth}(\mathcal{A})$ and $Adv_{\Pi}^{key-ind}(\mathcal{A})$ are negligible functions of the security parameter.

C. Security Proof

Without loss of generality, we consider the condition $\delta_{AB} = -1$ to prove SAKE* protocol under Brzuska et al.'s model Brzuska et al.'s model as described in [20]. We adopt security definitions of PRF and MAC in [17] to support the security proof.

***Theorem** 2:* The SAKE* protocol is secure since for any PPT adversary $\mathcal{A}$ against proposed SAKE*, we have

$$\begin{aligned} Adv_{SAKE*}^{auth}(\mathcal{A}) \leq & nl((nl-1)2^{-\lambda} + (l-1)Adv_{hash}^{SUF-CMA}(\mathcal{B}) + \\ & 2Adv_{MAC}^{SUF-CMA}(\mathcal{D}) + (l+1)Adv_{update}^{PRF}(\mathcal{E})) \end{aligned} \tag{1}$$

$$\begin{aligned} Adv_{\Pi}^{key-ind}(\mathcal{A}) \leq & nl((l-1)Adv_{update}^{PRF}(\mathcal{E}) + Adv_{KDF}^{PRF}(\mathcal{F})) \\ & + Adv_{SAKE*}^{auth}(\mathcal{A}) \end{aligned} \tag{2}$$

in which n is the maximum of entities, l the maximum of sessions each entity could execute, λ is the size of pseudo-random values (r_A, r_B), and $\mathcal{B}$ is the adversary against the security of $hash$, $\mathcal{D}$ an adversary fighting against Mac's SUF-CMA security, and $\mathcal{F}$ an adversary fighting against KDF's PRF-security.

Proof 2: Game 0. This game is the same as SAKE* protocol, which honestly executes entity authentication. Therefore

$$Pr[G_0] = Adv_{SAKE*}^{auth}(\mathcal{A})$$

Game 1. The challenger $\mathcal{C}$ will abort in this game if there is one instance where the random nonce r_A or r_B is exactly used in another different instance. Since the possible number of random values is $n \times l$, each is sampled randomly from $\{0, 1\}^\lambda$. Thereby, the maximum probability that there exists two random values are equal is $\frac{nl(nl-1)}{2^\lambda}$. Thus

$$Pr[G_0] \leq Pr[G_1] + \frac{nl(nl-1)}{2^\lambda}$$

Game 2. The challenger $\mathcal{C}$ attempts to guess the instance that is maliciously accepted in this game. If he wrongly guesses, this game is terminated. Due to the maximum of instances being nl. Thereby, it has

$$Pr[G_2] = Pr[G_1] \times \frac{1}{nl}$$

Game 3. In this game, we consider the maliciously accepted instance π^* where the messages $\{\epsilon_A, S_A, \tau_{CA}\}$ or $\{\epsilon_B, S_B, \tau_{CB}\}$ are successfully forged. We replace each update of evolution K_A (resp. K_B) with pseudo-random keyed hash function $h(K,\cdot)$. When the vth session begins, the evolution keys have been updated $v-1$ times. Since there are at most l session, the total loss is at most $(l-1)Adv_{hash}^{SUF-CMA}(\mathcal{B})$ during replacement. Moreover, computation of S_A, τ_{CA} (resp. S_B, τ_{CB}) depend on update of evolution keys, which means successful replacement of K_A (resp. K_B) achieves acceptable S_A (resp. S_B). Therefore, the probability of $\mathcal{A}$ winning this game is equal to the ability of $\mathcal{D}$ forging a valid τ_{CA} (resp. τ_{CB}). We have

$$Pr[G_2] \leq Pr[G_3]+(l-1)Adv_{hash}^{SUF-CMA}(\mathcal{B})+Adv_{MAC}^{SUF-CMA}(\mathcal{D})$$

Game 4. The challenger terminates in this game if π^* has received a valid message τ_{BA} (resp. τ_{AB}), but no instance has a matching session to π^* that has output that message. We substitute for each $update(K)$ using truly random functions $F_0^{update}, ..., F_{l-2}^{update}$. Absolute synchronization of master key K and evolution key K_A/K_B decides that K has updated at most $l-1$ times during τ_{BA} computation process. In addition, the master key update gap between τ_{CA} (resp. τ_{CB}) and τ_{BA} (resp. τ_{AB}) is at most twice update. Hence, we have

$$Pr[G_3] \leq Pr[G_4]+(l+1)Adv_{update}^{PRF}(\mathcal{E})+Adv_{Mac}^{SUF-CMA}(\mathcal{D})$$

Eventually, the unique way to make π^* maliciously accept is to transmit a valid τ_{BA} (resp. τ_{AB}) that was not the output of any other instance, in which case the challenger aborts. Thus $Pr[G_4] = 0$.

According to Game 0 to 4, we get the final inequation 1.

The key indistinguishability security is proved in the following part. Let E_i denote the event in which the adversary wins the key indistinguishability experiment in Game i, the advantage of which is $Adv_i = Pr[E_i] - \frac{1}{2}$.

Game 0. This game simulates the actual key indistinguishability experiment, and we have

$$Pr[E_0] = \frac{1}{2} + Adv_{SAKE*}^{key-ind}(\mathcal{A}) = \frac{1}{2} + Adv_0$$

Game 1. The challenger $\mathcal{C}$ will terminate in this game. $\mathcal{C}$ randomly selects $b' \in \{0,1\}$ if there is a maliciously accepted instance. Furthermore, we make the same replacement as those during the entity authentication proof. Thus, we have

$$Adv_0 \leq Adv_1 + Adv_{SAKE*}^{auth}(\mathcal{A})$$

Game 2. The challenger $\mathcal{C}$ attempts to guess the targeted instance of the adversary in this game. $\mathcal{C}$ aborts if the guess is wrong. They are at most nl instances. Thus we have

$$Adv_2 = Adv_1 \times \frac{1}{nl}$$

Game 3. Assume that $\pi*$ is the targeted instance of the adversary, the adversary's advantage of winning this game can be reduced to the KDF's security, where KDF is utilized to compute the session key. We replace $update(K) = PRF(K,x)$ with truly random functions $G_0^{update}, ..., G_{q-2}^{update}$. Due to each entity has at most l sessions, this loss is at most $(l-1)Adv_{update}^{PRF}$. Moreover, adversary success is reduced to the KDF's security. Thereby, we have

$$Adv_2 \leq Adv_3 + (l-1)Adv_{update}^{PRF}(\mathcal{E}) + Adv_{KDF}^{PRF}(\mathcal{F})$$

In this game, the negotiated key is completely random, and adversary $\mathcal{A}$ has no advantage to guess the right bit. That is $Adv_3 = 0$. According to Game 0 to 3, we get the final result 2.

D. Security Analysis

This subsection explains that the SAKE* protocol satisfies the following security properties.

- *Mutual authentication*: The main idea is that TA authenticates U_A and IoT_B through evolution keys K_A, K_B while both entities believe TA. Therefore, as long as U_A (resp. IoT_B) gets the valid MAC τ_{CA} (resp. τ_{CB}), he trust the other party IoT_B (resp. U_A) is legal, who has passed TA's verification. Theorem 2 formally proves that any PPT adversary cannot forge valid messages to deceive honest entities. Thus, the proposed SAKE* supports mutual authentication.
- *End-to-end security*: Since only IoT_B (resp. U_A) can read the current evolution key K_B (resp. K_A) from τ_{CA} (resp. τ_{CB}) and updated evolution key from S_A, which is ensured by security of MAC. In addition, nobody could read the session key authentication message τ_{BA} except for U_A. Therefore, SAKE* realizes end-to-end security.
- *Authenticated data integrity*: The data stream $\{ID_A, \epsilon_A, S_A, \tau_{CA}, ID_B, \epsilon_B, S_B, \tau_{CB}\}$ and $\{ID_B, r_B, \tau_{BA}, \epsilon_A, S_A, \tau_{CA}\}$ separately flow to IoT_B and U_A includes message authentication codes τ_{CB}, τ_{CA} and τ_{BA} to verify messages integrity. Any malicious modification can be found and causes verification to fail.
- *Secure session key agreement*: From the specifications of AKE phase, an honest IoT_B can produce a session key $sk = KDF(K, ID_A||ID_B||f(r_A, r_B))$ (kdf) after confirming U_A's validity and a legal U_A also can generate a correct sk in the SKAE* protocol. Moreover, consistence of sk is verified by $\tau_{BA} = Mac(K, r_A||r_B||sk)$. Theorem 2 has proved that probability of a PPT adversary breaking session key indistinguishability is negligible. Above all, the proposed SAKE* achieves secure key agreement.
- *Soundness*: When IoT_B receives his last messages flow, he updates master key K twice ($\epsilon_B = 1$) or once ($\epsilon_B = 0$). Meanwhile, he puts in τ_{BA} the generated session key for U_A's further authentication. The condition of U_A's master key update is the same as that of IoT_B. In addition, U_A uses τ_{BA} to ensure the negotiated session key is consistent with IoT_B's. Therefore, soundness is guaranteed.
- *Perfect forward secrecy*: The session key is computed by $sk = KDF(K, ID_A||ID_B||f(r_A, r_B))$ where K is master key. Soundness ensures K is updated at least once after each session. The irreversibility of the $update(\cdot)$ function means that previous master key cannot be recovered even if the present master key is disclosed. Thereby, an attacker cannot compute the previous session key using the captured master key.
- Known attacks resistance: we analyze the SAKE* protocol's resistance to the following five attacks.

- *Impersonation attack resilience*: In the SAKE* protocol, an impersonation attack means an adversary $\mathcal{A}$ could impersonate U_A/IoT_B to successfully the other party. If a malicious $\mathcal{A}$ attempts to impersonate U_A, he must successfully obtain U_A's current evolution key K_A to compute a valid τ_A. One way is that $\mathcal{A}$ uses $\tau_A = H(ID_A||PW_A||\theta_A) \oplus K_A$ in the public channel to get K_A. Nevertheless, he cannot know PW_A and θ_A because they are always kept secret from others, so it is difficult for $\mathcal{A}$ to steal PW_A, θ_A. The other way is that $\mathcal{A}$ directly steals K_A from storing devices. However, the evolution key is stored in a highly personal device under the symmetric-key mechanism. Therefore, $\mathcal{A}$ cannot forge a valid τ_A to impersonates U_A. This analysis is also applied to impersonating IoT_B attack. Moreover, $\mathcal{A}$ cannot forge τ_{CA} and τ_{BA} without knowing evolution key K_A^j and master key K. Above all, the proposed protocol can resist impersonation attack.
- *Man-in-the-middle attack resilience*: When an adversary $\mathcal{A}$ performs man-in-the-middle attack to SAKE*, he should firstly capture $\{ID_A, r_A, \tau_A\}$ from U_A and forge a valid $\{ID_A, r_A, \tau_A'\}$ to IoT_B. Meanwhile, $\mathcal{A}$ captures $\{\epsilon_A, S_A, \tau_{CA}, ID_B, r_B, \tau_{BA}\}$ and forges a valid $\{\epsilon_A', S_A', \tau_{CA}', ID_B, r_B', \tau_{BA}'\}$ to U_A. We have discussed that the SAKE* protocol withstands impersonation attack, thus $\mathcal{A}$ cannot forge valid messages to achieve man-in-the-middle attack.
- Replay attack resilience: For the first message flow $\{ID_A, r_A, \tau_A\}$ to IoT_B, IoT_B uses held two random values to verify whether it has appeared in recent two sessions and the TA only accepts twice delay of evolution keys compared with those he keeps. For the second message flow to IoT_B, $\tau_{CB} = Mac(K_2||\epsilon_B||K_B^j)$ where K_2 is IoT_B's current evolution key. Otherwise, it can cause τ_{CB} verification to fail. Similarly, τ_{CA} ensures that replay messages cannot succeed in U_A's verification. Therefore, replay attack resilience is provided.
- Eavesdropping attack resilience: We have discussed that nothing could be obtained from τ_A, τ_B since evolution keys K_A, K_B, passwords PW_A, PW_B and random numbers θ_A, θ_B are secretly kept, and the hash function used to generate g_A, update function of evolution keys are secure. Moreover, nobody could get master key K by τ_{CA}, τ_{BA} which is computed by secure MAC.
- Data tampering attack resilience: Since τ_{CA}/τ_{CB} contains all the transmitted information $(ID_A, \epsilon_A, S_A)/(ID_B, \epsilon_B, S_B)$ and τ_{CA}/τ_{CB} includes $(\epsilon_A, S_A)/(ID_B, r_B)$. Thus, any malicious alteration could make verification fail.

V. Security and Performance Comparisons

In this section, we compare our SAKE* protocol with seven recent and IoT-related protocols [24]–[30] in terms of twelve security properties, computation cost and communication cost of authentication and key exchange phase.

A. Security comparisons

As shown in table I, the proposed SAKE* provides all the security properties while other seven protocols does not guarantee perfect forward secrecy, formal security proof, replay attack resilience or impersonation attack resilience. Wu et al.'s protocol [24] cannot guarantee perfect forward secrecy and formal security proof. When an attacker gets the master key x, he can firstly guess the low-entropy identity SID_j through the equality $D_7 = h(ID_6 \oplus h(ID_g||h(SID_j||x)||T_2)||h(SID_j||x)||SID_j)$. Secondly, r_u could be computed by $D_6 \oplus h(ID_g||h(SID_j||x)||T_2)$ and r_s is computed by $D_8 \oplus r_u$. Finally, the attacker computes previous session key $SK = h(r_u||r_s)$.

Alsahlani-Popa's protocol [25] cannot provide perfect forward secrecy. Upon getting master key K_i, K_j, low-entropy ID_{U_i}, ID_{S_j} could be guessed by the equation $D_{11} = h(\gamma_4||h(ID_{S_j}||K_j)||D_9 \oplus h(\gamma_3||T_3)||T_3)$ and the SK_{ij} can be computed. Pham-Dang's protocol [26] cannot resist replay attack and impersonation attack since an attacker could utilize previous authenticated messages to replay and impersonate authenticated parties. These two attacks also exist in Li et al.'s protocol [28]. Moreover, Pham-Dang's protocol [26] also does not provide formal security proof.

Chaudhry et al.'s [27] cannot provide perfect forward secrecy. We can infer previous session key SK_{ij} with public $M_3, M_{10}, M_{12}, T_1, T_3$ when the master keys K_i, K_j are captured. The M_{12} can be represented by ID_{U_i}, ID_{S_j} where other values are known. Thereby, session key SK_{ij} could be recovered after low-entropy identities are guessed. With similar analysis method with that of [27], Wazid et al.'s [29] also does not guarantee perfect forward secrecy. Finally, Chatterjee et al.'s [30] does not provide formal security proof.

B. Computation costs comparison

We evaluate the computation overhead of the authentication phase by counting the number of significant cryptographic operations, such as scalar multiplication, the general hash function, AES encryption/decryption, and fuzzy extractor operation. Based on Miracl library [31], we test the running time of three operations on a personal computer and Raspberry Pi separately to simulate authenticated parities A&B and the TA. The running time of fuzzy extractor operation is quoted by [25]. Table II shows the running time of the four most consuming cryptographic operations.

Table III detailedly shows three parties' computational costs and total performing time comparisons of authentication and key exchange (AKE) phase. The analysis result indicates that SAKE* protocol has the least time consumption 0.1 ms in Side A among ten protocols, less time consumption 0.12 ms than other six protocols [17], [25]–[29] in Side B, less time consumption 0.036 ms than other six protocols [24], [26]–[30] in Side TA. Above all, the proposed SAKE* protocol has the lest total time consumption 0.256 ms of AKE phase.

TABLE I
COMPARISONS OF SECURITY PROPERTIES

Properties	Ref. [24]	Ref. [25]	Ref. [26]	Ref. [27]	Ref. [28]	Ref. [29]	Ref. [30]	Ours
P1	✓	✓	✓	✓	✓	✓	✓	✓
P2	✓	✓	✓	✓	✓	✓	✓	✓
P3	✓	✓	✓	✓	✓	✓	✓	✓
P4	✓	✓	✓	✓	✓	✓	✓	✓
P5	×	×	✓	×	✓	×	✓	✓
P6	-	-	-	-	-	-	-	✓
P7	✓	✓	×	✓	×	✓	✓	✓
P8	✓	✓	✓	✓	✓	✓	✓	✓
P9	✓	✓	×	✓	×	✓	✓	✓
P10	×	✓	✓	✓	✓	✓	✓	✓
P11	✓	✓	✓	✓	✓	✓	✓	✓
P12	×	✓	×	✓	✓	✓	×	✓

P1: Mutual authentication; P2: End-to-end security; P3: Transmitted data integrity; P4: Secure session key agreement; P5: Perfect forward secrecy; P6: Soundness; P7: Impersonation attack resilience; P8: Man-in-the-middle attack resilience; P9: Replay attack resilience; P10: Eavesdropping attack resilience; P11: Data tampering attack resilience; P12: Formal security proof.
✓: The security property is supported, or the attack can be withstood.
×: The security is not provided, or the attack cannot be withstood.
-: The security property is not mentioned or involved.

TABLE II
RUNNING TIME OF CRYPTOGRAPHIC OPERATIONS (MS)

Symbols	Descriptions (Running time)	Raspberry Pi	Computer
T_{sm}	A scalar multiplication	1.548	0.376
T_h	A hash function	0.02	0.003
T_s	AES encryption/decryption	0.142	0.05
T_{fe}	A fuzzy extractor	2.226	-

C. Communication costs comparison

We utilize $|P|$, $|Z_q^*|$, $|H|$, $|T|$ and $|ID|$ to separately denote lengths of an elliptic curve point, a larger integer, a hash function, a timestamp and an identifier, which are accordingly 512 bits, 160 bits, 256 bits, 64 bits and 32 bits to achieve the 1024-bits RSA algorithm security level. In the AKE phase of the proposed SAKE* protocol, U_A first sends a hash function output value τ_A, a random number r_A from $\{0,1\}^k$ to IoT_B. Secondly, IoT_B sends two hash values τ_A, τ_B and two identifiers ID_A, ID_B to the TA. Thirdly, TA transmits four hash values $S_A, S_B, \tau_{CA}, \tau_{CB}$, two identifiers ID_A, ID_B and two bits ϵ_A, ϵ_B to IoT. Finally, IoT_B sends three hash values $S_A, \tau_{CA}, \tau_{BA}$, a random number r_B, an identifier ID_B and one bit ϵ_A to U_A. Therefore, we have the communication costs that $|H| + |Z_q^*| + |ID|$ in A to B process, $2|H| + 2|ID|$ in B to TA process, $4|H| + 2|ID| + 2bits$ in TA to B process and $3|H| + |Z_q^*| + |ID| + 1bit$ in TA to A process. We evaluate the communication costs from the perspective of participated parties as depicted in figure 2.

We can conclude that our protocol consumes the least communication cost in the resource-limited Side A among these eight protocols, consumes fewer communication costs than Pham and Dang's [26], Chaudhry et al.'s [27] in Side B, and consumes fewer communication costs than Wu et al.'s [24], Li et al.'s [28], Chatterjee et al.'s [30] in the side TA. Above all, the communication cost of our protocol is the third least among the eight protocols.

VI. CONCLUSION

This paper designs an authenticated key exchange protocol for IIoT environments. The proposed SAKE* protocol guaranteed perfect forward secrecy in the symmetric cryptography mechanism and provided lightweight because of trust TA participation and simple XOR/hash function operations. We evaluated SAKE*'s security and performance to depict utility. The assessment is executed by comparing with seven IoT AKE-related protocols, which demonstrates that the proposed protocol supports secure mutual authentication, secure session key agreement, soundness, perfect forward secrecy, Etc. and withstands various security attacks. Furthermore, the SAKE* is proved secure under Brzuska et al.'s model, ensuring security in theory. The performance analysis and comparison indicate that the SAKE* is more applied to the IIoT environment than the known and existing AKE protocols.

REFERENCES

[1] F. Piccialli, N. Bessis, and E. Cambria, "Guest editorial: Industrial internet of things: Where are we and what is next?" *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7700–7703, 2021.

[2] G. Rathee, F. Ahmad, R. Iqbal, and M. Mukherjee, "Cognitive automation for smart decision-making in industrial internet of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 2152–2159, 2021.

[3] Y. Liu, X. Ma, L. Shu, G. P. Hancke, and A. M. Abu-Mahfouz, "From industry 4.0 to agriculture 4.0: Current status, enabling technologies, and research challenges," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4322–4334, 2021.

[4] M. N. Aladwan, F. M. Awaysheh, S. Alawadi, M. Alazab, T. F. Pena, and J. C. Cabaleiro, "Truste-vc: Trustworthy evaluation framework for industrial connected vehicles in the cloud," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6203–6213, 2020.

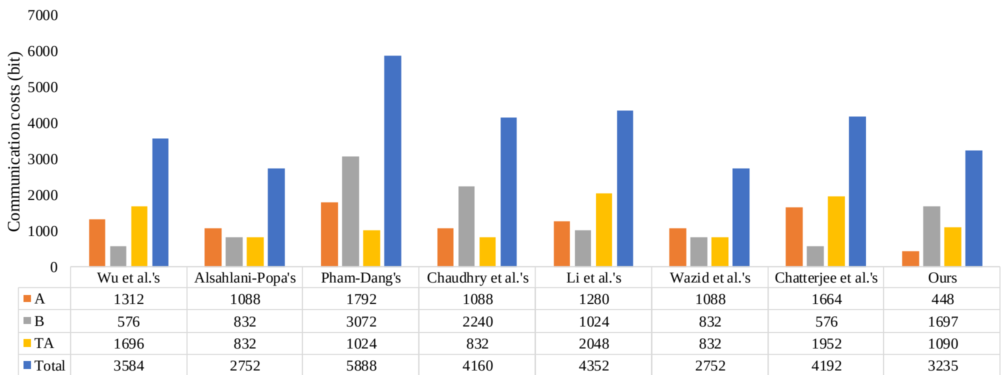
[5] J. M. Tim Hahn, Marcel Kisch, "Securing the internet of things for industrial and utility companies," https://www.ibm.com/downloads/cas/ZJRRVRKW.

[6] M. Abdel-Basset, V. Chang, H. Hawash, R. K. Chakrabortty, and M. Ryan, "Deep-ifs: Intrusion detection approach for industrial internet of things traffic in fog environment," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7704–7715, 2021.

[7] Y. Yang, X. Zheng, X. Liu, S. Zhong, and V. Chang, "Cross-domain dynamic anonymous authenticated group key management with symptom-matching for e-health social system," *Future Generation Computer Systems*, vol. 84, pp. 160–176, 2018.

TABLE III
COMPUTATION COSTS COMPARISON OF AKE PHASE (MS)

Protocols	Side A	Side B	TA	Total Time
Wu et al.'s [24]	$T_{fe} + 13T_h = 2.468$	$4T_h = 0.08$	$15T_h = 0.045$	2.593
Alsahlani and Popa's [25]	$T_{fe} + 14T_h = 2.506$	$7T_h = 0.14$	$4T_h = 0.012$	2.658
Pham and Dang's [26]	$3T_{sm} + 4T_s + 5T_h = 5.312$	$2T_{sm} + 14T_s = 5.084$	$T_{sm} + 14T_s + 3T_h = 1.166$	11.562
Chaudhry et al.'s [27]	$T_{fe} + 20T_h = 2.226$	$10T_h = 0.2$	$14T_h = 0.042$	2.468
Li et al.'s [28]	$3T_{sm} + 8T_h = 4.804$	$2T_{sm} + 4T_h = 3.146$	$T_{sm} + 8T_h = 0.4$	8.35
Wazid et al.'s [29]	$T_{fe} + 17T_h = 2.556$	$9T_h = 0.18$	$8T_h = 0.024$	2.76
Chatterjee et al.'s [30]	$11T_h = 0.22$	$5T_h = 0.1$	$17T_h = 0.051$	0.371
Avoine et al.'s [9]	$> 8T_h = 0.16$	$5T_h = 0.1$	-	>0.26
Avoine et al.'s [17]	$10T_h = 0.2$	$7T_h = 0.14$	-	0.34
Ours	$5T_h = 0.1$	$6T_h = 0.12$	$12T_h = 0.036$	0.256



A: The side of user.
B: The side of IoT device.
TA: Trust Authority.

Fig. 2. Communication costs comparison of AKE phase

[8] A. B. Rabiah, K. Ramakrishnan, E. Liri, and K. Kar, "A lightweight authentication and key exchange protocol for iot," in *Workshop on Decentralized IoT Security and Standards*, vol. 2018, 2018, pp. 1–6.

[9] G. Avoine, S. Canard, and L. Ferreira, "Iot-friendly ake: Forward secrecy and session resumption meet symmetric-key cryptography," in *Computer Security – ESORICS 2019*, K. Sako, S. Schneider, and P. Y. A. Ryan, Eds. Cham: Springer International Publishing, 2019, pp. 463–483.

[10] D. Fang, Y. Qian, and R. Q. Hu, "A flexible and efficient authentication and secure data transmission scheme for iot applications," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3474–3484, 2020.

[11] P. C. v. O. Whitfield Diffie and M. J. Wiener, "Authentication and authenticated key exchanges," *Designs, Codes and Cryptography*, vol. 2, no. 2, pp. 107–125, 1992.

[12] W. Wang, H. Huang, L. Zhang, and C. Su, "Secure and efficient mutual authentication protocol for smart grid under blockchain," *Peer-to-Peer Networking and Applications*, pp. 1–13, 2020.

[13] H. Xiong, C. Jin, M. Alazab, K.-H. Yeh, H. Wang, T. R. R. Gadekallu, W. Wang, and C. Su, "On the design of blockchain-based ecdsa with fault-tolerant batch verication protocol for blockchain-enabled iomt," *IEEE Journal of Biomedical and Health Informatics*, 2021.

[14] W. Wang, C. Qiu, Z. Yin, G. Srivastava, T. R. Gadekallu, F. Alsolami, and C. Su, "Blockchain and puf-based lightweight authentication protocol for wireless medical sensor networks," *IEEE Internet of Things Journal*, 2021.

[15] S. Garg, K. Kaur, G. Kaddoum, and K.-K. R. Choo, "Toward secure and provable authentication for internet of things: Realizing industry 4.0," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4598–4606, 2020.

[16] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Advances in Cryptology — CRYPTO' 93*, D. R. Stinson, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1994, pp. 232–249.

[17] G. Avoine, S. Canard, and L. Ferreira, "Symmetric-key authenticated key exchange (sake) with perfect forward secrecy," in *Topics in Cryptology – CT-RSA 2020*, S. Jarecki, Ed. Cham: Springer International Publishing, 2020, pp. 199–224.

[18] M. Bellare, R. Canetti, and H. Krawczyk, "Keying hash functions for message authentication," in *Advances in Cryptology — CRYPTO '96*, N. Koblitz, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 1–15.

[19] H. Krawczyk, "Cryptographic extraction and key derivation: The hkdf scheme," in *Advances in Cryptology – CRYPTO 2010*, T. Rabin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 631–648.

[20] C. Brzuska, H. Jacobsen, and D. Stebila, "Safely exporting keys from secure channels," in *Advances in Cryptology – EUROCRYPT 2016*, M. Fischlin and J.-S. Coron, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 670–698.

[21] S. Li, T. Zhang, B. Yu, and K. He, "A provably secure and practical puf-based end-to-end mutual authentication and key exchange protocol for iot," *IEEE Sensors Journal*, vol. 21, no. 4, pp. 5487–5501, 2021.

[22] M. Bellare and P. Rogaway, "Entity authentication and key distribution," in *Annual international cryptology conference*. Springer, 1993, pp. 232–249.

[23] N. A. Anagnostopoulos, S. Ahmad, T. Arul, D. Steinmetzer, M. Hollick, and S. Katzenbeisser, "Low-cost security for next-generation iot networks," *ACM Trans. Internet Technol.*, vol. 20, no. 3, Sep. 2020. [Online]. Available: https://doi.org/10.1145/3406280

[24] F. Wu, X. Li, L. Xu, P. Vijayakumar, and N. Kumar, "A novel three-factor authentication protocol for wireless sensor networks with iot notion," *IEEE Systems Journal*, vol. 15, no. 1, pp. 1120–1129, 2021.

[25] A. Y. F. Alsahlani and A. Popa, "Lmaas-iot: Lightweight multi-factor authentication and authorization scheme for real-time data access in iot cloud-based environment," *Journal of Network and Computer Applications*, vol. 192, p. 103177, 2021.

[26] C. D. Pham and T. K. Dang, "A lightweight authentication protocol for d2d-enabled iot systems with privacy," *Pervasive and Mobile Computing*, vol. 74, p. 101399, 2021.

[27] S. A. Chaudhry, A. Irshad, K. Yahya, N. Kumar, M. Alazab, and Y. B. Zikria, "Rotating behind privacy: An improved lightweight authentication scheme for cloud-based iot environment," *ACM Trans. Internet Technol.*, vol. 21, no. 3, Jun. 2021.

[28] X. Li, J. Peng, M. S. Obaidat, F. Wu, M. K. Khan, and C. Chen, "A secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems," *IEEE Systems Journal*, vol. 14, no. 1, pp. 39–50, 2020.

[29] M. Wazid, A. K. Das, V. Bhat K, and A. V. Vasilakos, "Lam-ciot: Lightweight authentication mechanism in cloud-based iot environment," *Journal of Network and Computer Applications*, vol. 150, p. 102496, 2020.

[30] A. Ostad-Sharif, H. Arshad, M. Nikooghadam, and D. Abbasinezhad-Mood, "Three party secure data transmission in iot networks through design of a lightweight authenticated key agreement scheme," *Future Generation Computer Systems*, vol. 100, pp. 882–892, 2019.

[31] M. Ltd., "Miracl cryptographic sdk: Multiprecision integer and rational arithmetic cryptographic library," https://github.com/miracl/MIRACL.

Qing Fan is a PhD student in applied mathematics from Wuhan University, Wuhan, China.

Jianhua Chen received the M.S. and Ph.D. degrees in applied mathematics from Wuhan University, Wuhan, China, in 1989 and 1994, respectively. He is currently a Professor with Wuhan University. His current research interests include number theory, information security, and network security.

Mohammad Shojafar (M'17-SM'19) is a Senior Lecturer (Associate Professor) in the network security and an Intel Innovator, Fellow of Higher Education Academy, Professional ACM member and ACM Distinguished Speaker, and a Marie Curie Alumni, working in the 5G & 6G Innovation Centre (5GIC & 6GIC), Institute for Communication Systems (ICS), at the University of Surrey, UK. Before joining 5GIC/6GIC, he was a Senior Researcher and a Marie Curie Fellow in the SPRITZ Security and Privacy Research group at the University of Padua, Italy. Also, he was CNIT Senior Researcher at the University of Rome Tor Vergata contributed to 5G PPP European H2020 "SUPERFLUIDITY" project. Dr. Mohammad is a PI of AutoTrust, a 750k euro 5G secure autonomous vehicular communication project supported by European Space Agency (ESA) in 2021 and was a PI of PRISENODE project, a 275k euro Horizon 2020 Marie Curie global fellowship project in the areas of Fog/Cloud security collaborating at the University of Padua, Italy. He also was a PI on an Italian SDN security and privacy (60k euro) supported by the University of Padua in 2018. He was contributed to some Italian projects in telecommunications like GAUChO, SAMMClouds, and SC2. He received his Ph.D. degree in ICT from Sapienza University of Rome, Rome, Italy, in 2016 with an "Excellent" degree. He is an Associate Editor in *IEEE Transactions on Network and Service Management*, *IEEE Transactions on Consumer Electronics*, *IEEE Systems Journal* and Computer Networks. For additional information: http://mshojafar.com

Saru Kumari is currently an Assistant Professor with the Department of Mathematics, Chaudhary Charan Singh University, Meerut, Uttar Pradesh, India. She received her Ph.D. degree in Mathematics in 2012 from Chaudhary Charan Singh University, Meerut, UP, India. She has published more than 215 research papers in reputed International journals and conferences, including more than 195 research papers in various SCI-Indexed Journals. She is on the editorial board of more than a dozen of International Journals, of high repute, under IEEE, Elsevier, Springer, Wiley and others including IEEE Systems Journal. She has served as the Guest Editor of many special issues in many SCI Journals under IEEE, Elsevier, Springer and Wiley. She has been involved in the research community as Technical Program Committee (TPC) member or PC chair for more than a dozen of International conferences of high repute. She is also serving as a reviewer of dozens of reputed Journals including SCI-Indexed of IEEE, Elsevier, Springer, Wiley, Taylor & Francis, etc. Her current research interests include information security and applied cryptography.

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University, Wuhan, China in 2009. He is currently a professor of the School of Cyber Science and Engineering, Wuhan University, Wuhan, China. His main research interests include cryptography and information security, in particular, cryptographic protocols. He has published over 100 research papers in refereed international journals and conferences, such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Security and Forensic, and Usenix Security Symposium. He is the recipient of the 2018 IEEE Sysems Journal Best Paper Award and the 2019 IET Information Security Best Paper Award. His work has been cited more than 10000 times at Google Scholar. He is in the Editorial Board of several international journals, such as Journal of Information Security and Applications, Frontiers of Computer Science, and Human-centric Computing & Information Sciences.